ACADEMIC PRESS

Available at www.ComputerScienceWeb.com
POWERED BY SCIENCE @ DIRECT®

Information and Computation 182 (2003) 53–71

Information and Computation

www.elsevier.com/locate/ic

Syntactic analysis of η-expansions in Pure Type Systems

Felix Joachimski

Mathematisches Institut, Ludwig-Maximilians-Universität, Theresienstrasse 39, 80333 München, Germany

Received 22 February 2002; revised 31 May 2002

Abstract

By a detailed analysis of the interaction between β-reduction $\rightarrow_\beta$ and η-expansion $\rightarrow_{\bar{\eta}}$ in the simply typed λ-calculus, a modular and purely syntactic proof method is devised in order to derive strong normalization of the combined reduction $\rightarrow_{\beta\bar{\eta}}$ from that of $\rightarrow_\beta$ and $\rightarrow_{\bar{\eta}}$. It is shown how this technique extends to β-normalizing functional Pure Type Systems with Barthe's formulation of η-expansion.

Keywords: Pure type systems; η-Expansion; Strong normalization

1. Introduction

$\beta\eta$-equality for the λ-calculus is of major interest in programming language semantics for arguing about denotational equivalence and in proof theory for normal form analysis.

Decidability of $\beta\eta$-equality is often achieved by turning it into a reduction relation and comparing normal forms. Traditionally, η-equality $\lambda x.rx =_\eta r$ (if $x \notin \mathrm{FV}r$) has been oriented into η-contraction $\lambda x.rx \rightarrow_\eta r$ which models the computational aspect of extensionality. As pointed out by Prawitz [12], the expansive version $\rightarrow_{\bar{\eta}}$ (generated from $r \rightarrow_{\bar{\eta}} \lambda x.rx$) is better suited for analysis of normal forms. Other applications of η-expansions can be found in higher-order unification, partial evaluation and many other fields [5]. For the Calculus of Constructions, Dowek et al. [6] have analyzed the resulting notion of long normal forms; they also justified an induction principle along the structure of long normal forms.

η-Expansion is better formulated in the typed λ-calculus, for otherwise any term is expandable. Also, λ-abstractions should be prevented from being expanded in order that

E-mail address: felix@joachimski.de.

0890-5401/03/$ - see front matter
doi:10.1016/S0890-5401(03)00019-1

η-expansion preserve β-normal forms. As a consequence, η-expansion is not substitutive (y might expand to $\lambda x.yx$ whereas $y[y := \lambda xx] = \lambda xx$ should not expand to the non-β-normal term $\lambda x.(\lambda xx)x$). Appropriately restricted notions of η-expansion have been studied by Ghani [9] and [3], where they have been shown to be terminating for a large class of Pure Type Systems.

In arbitrary combination with β-reduction $\to_\beta$, η-expansion poses some difficulties, e.g., a term rs might expand to $(\lambda x.rx)s$ which can be β-reduced to rs again. To avoid such reduction loops, one has to prevent terms in applicative positions from being expanded, thus losing compatibility in addition to substitutivity. This in turn makes it notoriously difficult to argue about the combined reduction relation $\to_{\beta\bar{\eta}}$, e.g., in strong normalization proofs.

In the last 20 years various methods have been devised to overcome the problems in strong normalization proofs for combined β-reduction and η-expansion $\to_{\beta\bar{\eta}}$. A thorough account of this long history of strong normalization proofs and their classification has been given in [5]. The most advanced result has been attempted by Barthe in [4] where strong normalization for $\to_{\beta\bar{\eta}}$ in the Calculus of Constructions is conjectured. Unfortunately, the underlying model construction is rather complex and some steps in the proof are fragmentary, so that the final word on the method is not yet spoken.

As remarked in [4], a more modular result would be desirable in order to circumstantiate Barthe's extension of Geuvers' conjecture: In Pure Type Systems (and generalizations or extensions thereof) strong normalization of $\to_{\beta\bar{\eta}}$ in the system with $\beta\eta$-equality should follow from strong normalization of $\to_\beta$ in the system with β-equality.

In this work a concise syntactic analysis of the interaction between the two reduction relations $\to_\beta$ and $\to_{\bar{\eta}}$ is provided, leading to simple commutation properties which allow to derive the second part of (a variant of) Barthe's conjecture.

- In a first step a subset of β-reductions (dubbed *strict*) is isolated which cannot newly arise through η-expansion. They serve to construct a notion of reduction v that is invariant under inverse η-reduction. This part of the analysis applies already to the untyped λ-calculus, because it only involves commutation properties with backward η-reduction (which forms a superset of η-expansion).
- The second step uses types and establishes that the remaining non-strict β-reductions are limited in number, even if combined with η-expansions. Although types are needed to guarantee strong normalization of η-expansions, their structure is not important.
- As a consequence, the second step can be adapted to the general case of (functional) Pure Type Systems. The main obstacle there stems from reductions in types A which are abstracted by $\lambda x : A.r$. We will have to forbid η-expansions in such abstracted types in order to retain the commutation properties mentioned before.
- In the final step a simulation method (developed to embed conventional Pure Type Systems into domain-free ones) is invoked to derive strong normalization of the full reduction system, assuming a closure property of the rule set, which holds for many practical systems, including the Calculus of Constructions.

Taken together these steps yield strong normalization for $\to_{\beta\bar{\eta}}$, provided that $\to_{\beta\eta}$ and $\to_{\bar{\eta}}$ are strongly normalizing in the Pure Type System with $\beta\eta$-equality and the relevant abstraction rules exist. By reference to [7] and [3], the first two premises hold in β-normalizing functional Pure Type Systems.

Outline of the contents. Section 2 introduces the method for the simply typed λ-calculus, where already most of the combinatorial issues can be explained. Working in the untyped λ-calculus first, it introduces δ and v as auxiliary reduction relations and analyzes their concurrent behavior. After switching to the typed system, η-expansion is defined and the combination with β-reduction proved strongly normalizing. Section 3 works out the basic commutation properties that hold for raw terms of Pure Type Systems and, in particular, discusses the peculiarities of reduction in abstracted types. Section 4 recalls Barthe's definition of η-expansion and completes the strong normalization argument by means of a type assignment to δ-reduction and elaboration of the above mentioned simulation method.

Notation. Greek symbols ϕ denote binary relations. We write $r \to_\phi s$ for $(r,s) \in \phi$ in order to recover the usual formulation of reductions. The following notations will be used.[1]

ε	for equality
$\phi\psi$	for the union of ϕ and ψ
$\phi^\pm$	for $\varepsilon\phi$
$\phi \cap \psi$	for the intersection of ϕ and ψ
$\phi \setminus \psi$	for the set-theoretic subtraction $\phi \setminus \psi$
$\phi \rhd \psi$	for the relational composition $\psi \circ \phi$
ϕ^+	for the transitive closure of ϕ
ϕ^*	for the reflexive transitive closure of ϕ (i.e., $\varepsilon\phi^+$)
$\underline{\phi}$	for the inverse relation
ε_ϕ	for the equality induced by ϕ (i.e., $(\phi\underline{\phi})^*$)
$=_\phi$	for $\to_{\varepsilon_\phi}$
$\phi \Downarrow$	for "ϕ is strongly normalizing"
$r \Downarrow_\phi$	for "r is strongly normalizing w.r.t. ϕ"
$\#_\phi r$	for the height of the ϕ-reduction tree (defined only if $r \Downarrow_\phi$)
NF_ϕ	for the set of normal forms w.r.t. ϕ

We give $\cap$ the highest and $\rhd$ the lowest priority, so that, e.g., $\phi^*\psi \rhd \phi \cap \psi = (\phi \cap \psi) \circ (\phi^* \cup \psi)$.

Lists of the form $e_1, \ldots, e_n$ are written $\vec{e}$, including the case $\epsilon := \vec{e}$ if $n = 0$. We write $e, \vec{e}$ to prefix e to the list $\vec{e}$ and identify the one-element list e, ϵ with e.

2. η-Expansion in the simply typed λ-calculus

This section provides a strong normalization proof for $\beta\bar{\eta}$ relative to strong normalization for β and $\bar{\eta}$ alone. Conceptually, the method employed is much more involved than those to be found in the literature (see [4] for a list), but it yields an analysis of the basic behavior of η-expansion in relation to β-reduction and thus prepares the later treatment of Pure Type Systems, rendering it more admissible.

[1] Although these algebraic notations for operations on reduction relations seem rather technical, they illustrate nicely how one relation is permuted through another, see Lemma 2 in Section 2.3 for an example. Diagrammatic illustration of the respective manipulations on reduction sequences would require much more space than is available.

Note that the strong normalization problem is not trivial – although η-expansions are defined so as to prevent creation of new β-redexes (they preserve β-normal forms), an η-expansion can provoke β-reductions in the future; for instance

$$(*) \qquad (\lambda x \lambda yy)x \to_{\bar\eta} \lambda y.(\lambda x \lambda yy)xy \to_\beta \lambda y.(\lambda yy)y \to_\beta \lambda yy.$$

In the simply typed λ-calculus, η-expansion enjoys the commutation property $(\beta\bar\eta)^* \subset \bar\eta^* \triangleright \beta^*$, which allows proofs using simulation of β-reductions on $\bar\eta$-normal forms (as used, e.g., in [2]) or an adaptation of the customary strong normalization proofs by logical relations (as can be found for instance in [8]). Yet, the commutation property fails in more complex type systems such as system F, so that stronger extensions of saturated sets became necessary [10]. These methods, however, do not immediately translate to the Calculus of Constructions or other Pure Type Systems.

The core observation of this paper is that the newly arising β-reductions in $(*)$ are of a very simple form – in the example above they are just η-reductions. But before we can formulate the general form of such reductions, let us first fix the relevant λ-calculus notations.

2.1. Basic definitions

Terms are generated from an infinite supply of variables x, y, z by the grammar

$$\Lambda \ni r, s, t \ ::= \ x \,|\, \lambda xr \,|\, rs.$$

As usual, we let application associate to the left, writing $r\vec{s}$ for the iterated application of r to the elements of $\vec{s}$. The variable x is bound in λxr. We identify terms that only differ in names of bound variables, adopting the standard variable conventions. The dot-notation $\lambda x.r$ is used to omit the parentheses in $\lambda x(r)$, where the range of the dot extends as far to the right as syntactically possible. We write $r[x := s]$ for the capture free substitution and FVr for the set of free variables, both defined as usual.

Reduction. *β- and η-contraction* are given by

$$(\lambda xr)s \to_{\beta_c} r[x := s]. \qquad \lambda x.rx \to_{\eta_c} r \quad \text{if } x \notin \mathrm{FV}r.$$

β and η denote the full *term closures* of β_c and η_c (defined inductively along the term structure). If the reduction relation ϕ is derived as the term closure of ϕ_c then elements of $\phi \setminus \phi_c$ are called *inner reductions.*

The aim of the following subsections is to isolate and analyze the subset of β-reductions that arise indirectly through η-expansions, as exemplified in $(*)$. In fact, it is easier to first consider backward η-reduction $\underline{\eta}$ (which forms a superset of η-expansion). It turns out that the β-reductions provoked by $\underline{\eta}$ are of two types. One is given by $\eta_c \cap \beta$, reducing $\lambda x.(\lambda xr)x$ to λxr. The other is an appropriate generalization of $(\lambda x.rx)s \to_{\eta \cap \beta_c} rs$, to be defined next.

2.2. Extended $\eta \cap \beta_c$-contraction

We define special terms Δ with one occurrence of a distinguished variable $\{\}$ (a "hole" into which we substitute (capture free) by writing $\Delta\{r\}$) inductively by

$$\delta \ni \Delta \ ::= \ \lambda x.\Delta_0\{\Delta_1\{\}\Delta_2\{x\}\} \,|\, \{\}.$$

We also write Δr for $\Delta\{r\}$. $\hat{\Delta}$ denotes a $\Delta \neq \{\}$. $(\Delta r)@s$ is defined only for $\hat{\Delta}$:

$$(\lambda x.\Delta_0(\Delta_1 r \Delta_2 x))@s := \Delta_0(\Delta_1 r \Delta_2 s).$$

Extended contraction Δ_c is given by $(\Delta r)s \to_{\Delta_c} (\Delta r)@s$. δ_c denotes the union of all Δ_c and δ stands for the term closure of δ_c.[2]

Examples for Δ of increasing complexity are

$$\lambda x.\{\}x, \quad \lambda x\lambda y.(\lambda z.\{\}z)xy, \quad \lambda x\lambda y.(\lambda z.\{\}z)(\lambda z.xz)y.$$

The corresponding δ-contractions are

$$\begin{aligned}(\lambda x.rx)s &\to_{\eta\cap\beta_c} rs,\\ (\lambda x\lambda y.(\lambda z.rz)xy)s &\to_{\delta_c} \lambda y.(\lambda z.rz)sy,\\ (\lambda x\lambda y.(\lambda z.rz)(\lambda z.xz)y)s &\to_{\delta_c} \lambda y.(\lambda z.rz)(\lambda z.sz)y.\end{aligned}$$

Notation. We write $\Delta \to_\phi \Delta'$ iff $\Delta x \to_\phi \Delta' x$ for an x (the choice is irrelevant, for all contraction rules used up to now are substitutive and Δ is closed).

Remarks.
1. By definition $\delta_c \subset \beta_c$ and therefore $\delta \subset \beta$.
2. $\Delta r \to_{\eta^*} r$. For $\hat{\Delta}$ we get $\hat{\Delta} \to_{\eta^+} r$. [Induction on Δ. *Case* $\{\}$. Trivial. *Case* $\lambda x.\Delta_0(\Delta_1\{\}\Delta_2 x)$: $\lambda x.\Delta_0(\Delta_1 r \Delta_2 x) \overset{\text{IH}}{\to}_{\eta^*} \lambda x.\Delta_0((\Delta_1 r)x) \overset{\text{IH}}{\to}_{\eta^*} \lambda x.\Delta_0(rx) \overset{\text{IH}}{\to}_{\eta^*} \lambda x.rx \to_{\eta_c} r$.]
3. A simple induction verifies that $\delta \subset \eta^* \rhd \eta \cap \beta_c \rhd \underline{\eta}^*$ – in that sense δ generalizes $\eta \cap \beta_c$.
4. δ_c is a contraction relation, i.e., it is substitutive

   $$r \to_\delta r' \Rightarrow r[x := s] \to_\delta r'[x := s]$$

   and does not create variables.
5. β- and η-reductions on either Δ, r or s can be simulated on $(\Delta r)@s$, except for an η-contraction in $\Delta = \lambda x.\{\}x$: $(\lambda x.rx)@s = rs = \{r\}s$.

Proposition 1. $\Delta r \to_{\eta\underline{\eta}} s \iff$

$$s = \Delta r' \quad \textit{with } r \to_{\eta\underline{\eta}} r' \qquad \textit{or} \qquad s = \Delta' r \quad \textit{with } \Delta \to_{\eta\underline{\eta}} \Delta'.$$

Proof. $\Rightarrow$: Induction on Δ.[3] $\Leftarrow$: By compatibility of η. $\square$

The next goal is to substantiate the idea that $\beta \cap \eta$- and δ-reductions are computationally irrelevant. The proof of strong normalization for $\beta\overline{\eta}$ relies on the fact that the number of possible computationally relevant β-reductions in a term is not changed by η-expansion, although new $\beta \cap \eta$- and δ-reductions might be introduced.

[2] Just as θ, γ and v (to be introduced below), the reduction relation δ is but a technical tool and bears no deeper relevance outside the scope of this paper.

[3] A similar assertion for η-expansion rather than $\underline{\eta}$ will later be proved in detail.

2.3. θ- and $\beta!$-reduction

We merge the computationally irrelevant reductions in $\theta := (\beta \cap \eta)\delta$ and define *strict β-reduction* to be the complement, i.e., $\beta! := \beta \setminus \theta$. Since $\beta \cap \eta \subset \theta \subset \beta$ the relation β decomposes into θ and $\beta!$. Only strict β-reductions are considered computationally relevant, for θ forms just a subset of ε_η.[4]

Later we will state precisely how θ- and $\beta!$-reductions interact with η-expansion. For the time being we concentrate on a superset of η-expansion: inverse η-reduction $\underline{\eta}$. The following lemma illustrates how $\beta!$-reductions are stable under permutation through $\underline{\eta}$-reduction whereas θ-reductions might disappear.

Lemma 2 (Commutation properties).[5]

(1) $\underline{\eta} \triangleright \beta! \subset \beta! \triangleright \underline{\eta}^*$,

(2) $\underline{\eta} \triangleright \beta \cap \eta \subset (\beta \cap \eta \triangleright \underline{\eta})\varepsilon$,

(3) $\underline{\eta} \triangleright \delta \subset (\delta \triangleright \underline{\eta})\varepsilon$.

Proof. (1) Assume $s_\eta \leftarrow r \rightarrow_{\beta!} t$. We show $\exists r'.s \rightarrow_{\beta!} r'_{\eta^*} \leftarrow t$ by induction on r and distinguish cases as follows.

Case $s_{\eta_c} \leftarrow r = \lambda x.sx$. In this case the strict β-reduction $r \rightarrow_{\beta!} t$ must be of the form $\lambda x.sx \rightarrow_{\beta!} \lambda x.s'x$, because if the reduction actually harmed the term structure $\lambda x.sx$, it would have the form $\lambda x.sx = \lambda x.(\lambda xs')x \rightarrow_{\beta\cap\eta} \lambda xs'$, contradicting strictness. Thus $s \rightarrow_{\beta!} s'$ and we can join $s \rightarrow_{\beta!} s'_{\eta_c} \leftarrow \lambda x.s'x$.

Case $(\lambda xr')s_\eta \leftarrow (\lambda xr)s \rightarrow_{\beta!} r[x := s]$. We would like to reduce

$$(\lambda xr')s \rightarrow_{\beta!} r'[x := s]_\eta \leftarrow r[x := s],$$

but need to guarantee strictness of the left reduction. Two objections to strictness are conceivable:

- What if $(\lambda xr')s \rightarrow_{\delta_c} r'[x := s]$? Then $(\lambda xr')s$ were of the form $(\Delta t)s$, so by the proposition, since $\lambda xr \rightarrow_\eta \Delta t$, also $\lambda xr = \Delta' t'$, contradicting strictness of $(\lambda xr)s \rightarrow_{\beta!} r[x := s]$.
- What if $(\lambda xr')s \rightarrow_{\beta\cap\eta} r'[x := s]$? Then $\lambda xr'$ were of the form $\lambda x.r''x$ with $x \notin \mathrm{FV}r''$. But $r \rightarrow_\eta r''x$ is only possible if $r = \hat{r}x$ with $x \notin \mathrm{FV}\hat{r}$, because η neither creates new applications nor does it lose variables.

Case $(\lambda xr)s'_\eta \leftarrow (\lambda xr)s \rightarrow_{\beta!} r[x := s]$. This can be joined by parallel substitutivity of $\rightarrow_{\eta^*}$:[6]

$$(\lambda xr)s' \rightarrow_{\beta!} r[x := s']_{\eta^*} \leftarrow r[x := s].$$

In the remaining cases neither reduction is a contraction, so they occur in proper subterms. If both reductions take place in the same subterm, we can apply the induction hypothesis. Otherwise the reductions are independent and can simply be executed.

[4] This simple fact might suggest that we are essentially arguing about β-reduction modulo η-equality. Such an intuitive approach unfortunately fails.

[5] The reader is invited to expand the algebraic notation of these and all following statements to their full existential formulation. For instance, property (2) reads $\forall r, s, t.r \rightarrow_{\underline{\eta}} s \rightarrow_{\beta\cap\eta} t \Rightarrow r = t \vee \exists s'.r \rightarrow_{\beta\cap\eta} s' \rightarrow_{\underline{\eta}} t$.

[6] For a substitutive contraction ϕ_c the transitive reflexive closure ϕ^* of the term closure ϕ is parallel-substitutive, i.e., $r \rightarrow_{\phi^*} r'$ & $s \rightarrow_{\phi^*} s' \Rightarrow r[x := s] \rightarrow_{\phi^*} r'[x := s']$.

(2) Assume $s_\eta \leftarrow r \rightarrow_{\beta\cap\eta} t$. We show $s = t \vee \exists r'.s \rightarrow_{\beta\cap\eta} r'_\eta \leftarrow t$ by induction on r.

Case both reductions are η-contractions. We are done, since $s = t$.

Case $s_{\eta_c} \leftarrow \lambda x.sx = r$. If $r = \lambda x.sx \rightarrow_{\eta_c} t = s$, we are in the previous case. Otherwise $t = \lambda x.s'x$ with $s \rightarrow_{\beta\cap\eta} s'$, so an η-contraction $t \rightarrow_{\eta_c} s'$ completes the join.

Case $r = \lambda x.(\lambda xr')x \rightarrow_{\eta_c} \lambda xr' = t$. If $r \rightarrow_{\eta_c} s$, we are in the first case, again. Otherwise $s = \lambda xs'$ with $(\lambda xr')x \rightarrow_\eta s'$. This reduction is either a combined $\beta\cap\eta$-reduction with $r' = r''x$ and we can complete the reduction by $t = \lambda xr' = \lambda x.r''x \rightarrow_{\eta_c} r''$. Or it is an inner reduction of $r' \rightarrow_\eta r''$ with $s' = (\lambda xr'')x$, so we join $t = \lambda xr' \rightarrow_\eta \lambda xr''{}_{\beta\cap\eta} \leftarrow \lambda x.(\lambda xr'')x = s$.

The remaining cases of two inner reductions are dealt with as in (1).

(3) Assume $s_\eta \leftarrow r \rightarrow_\delta t$. We show $s = t \vee \exists r'.s \rightarrow_\delta r'_\eta \leftarrow t$ by induction on r.

Case $s_{\eta_c} \leftarrow \lambda x.sx \rightarrow_\delta t$. The δ-reduction can only be an inner one of the form $\lambda x.sx \rightarrow_\delta t = \lambda xt'$. Now if $sx \rightarrow_\delta t'$ is a contraction $sx = (\Delta s')x \rightarrow_{\delta_c} (\Delta s')@x$ then we use $s = \lambda x.s@x$ (shown by a trivial induction). Otherwise the δ-contraction has to occur inside s, i.e., has the form $s \rightarrow_\delta r'$ and we reduce $s \rightarrow_\delta r'_\eta \leftarrow \lambda x.r'x$.

Case $s_\eta \leftarrow (\Delta r_0)r_1 \rightarrow_{\delta_c} t = (\Delta r_0)@r_1$. By the proposition s has one of the three forms

$$(\Delta' r_0)r_1 \quad \text{or} \quad (\Delta r_0')r_1 \quad \text{or} \quad (\Delta r_0)r_1'.$$

In the first subcase Δ' might be $\{\}$ due to $\Delta r = \lambda x.rx \rightarrow_{\eta_c} r$. But then $t = (\Delta r_0)@r_1 = r_0 r_1 = s$. Otherwise Δ' is not $\{\}$. We write $s = (\Delta' r_0')r_1'$ to subsume this and the two remaining subcases, and define the joining reduction by $s = (\Delta' r_0')r_1' \rightarrow_{\delta_c} (\Delta' r_0')@r'_{1\eta} \leftarrow (\Delta r)s$.

Case neither reduction is a contraction. Then argue as in (1). □

2.4. *ν-Reduction*

To get a notion of reduction that is invariant under η-expansion and θ-reduction we define $\nu := (\underline{\eta}\theta)^* \triangleright \beta!$. More verbosely, a ν-step consists of one strict β-reduction, preceded by a few backward η- or forward θ-steps.

Theorem 3. $\nu \subset \beta^+ \triangleright \underline{\eta}^*$.

Proof. Using the commutation properties (2) and (3) we can normalize any given ν-reduction $(\underline{\eta}\theta)^* \triangleright \beta!$ to the form $\beta^* \triangleright \underline{\eta}^* \triangleright \beta!$. By (1) we can commute $\beta!$ through the sequence $\underline{\eta}^*$, so that we obtain $\beta^* \triangleright \beta! \triangleright \underline{\eta}^*$. □

We can use this theorem to establish $\#_\nu$ as an induction measure which is invariant under $\underline{\eta}\theta$:

Corollary 4. $r \Downarrow_\beta \Rightarrow r \Downarrow_\nu$.

Proof. Induction on $r \Downarrow_\beta$. By the theorem, any ν-reduction $r \rightarrow_\nu t$ can be turned into one of the form $r \rightarrow_{\beta^+} s_{\eta^*} \leftarrow t$. By induction hypothesis $s \Downarrow_\nu$. Therefore $t \Downarrow_\nu$, since any ν-reduction $t \rightarrow_\nu t'$ can be prefixed with $s_{\eta^*} \leftarrow t \rightarrow_\nu t'$ and thus $s \rightarrow_\nu t'$. □

2.5. Types and η-expansion

Types ρ,σ,τ are generated from *basic types* ι by $\rho \to \sigma$. Given a unique assignment $x : \rho$ of types to variables such that for each type infinitely many variables exist, the *typable terms* and their unique types are inductively determined by the rules

$$\frac{r : \rho \to \sigma \quad s : \rho}{rs : \sigma} \qquad \frac{r : \sigma \quad x : \rho}{\lambda xr : \rho \to \sigma}.$$

We will decorate (sub-)terms with types in superscripts (as in $r^\rho s$) in order to signify that they are typable and get the respective type. $\Lambda^{\to}$ denotes the set of typable terms w.r.t. a type assignment for variables, that shall henceforward be fixed.

For the rest of this section we restrict our focus and all quantifiers and reduction relations to typable terms, i.e., to $\Lambda^{\to}$. We presuppose strong normalization and confluence for $\beta\eta$ (see [11] for a simple proof).

η-Expansion. A term is *neutral* if it is no abstraction λxr. *$\bar\eta$-expansion* is given by

$$r^{\rho\to\sigma} \to_{\bar\eta_c} \lambda x^\rho.rx \quad \text{if } r \text{ is neutral.}$$

Clearly, η-expansion makes sense only in non-applicative contexts. Therefore the notion of term closure has to be modified for $\bar\eta$:

$$\frac{r \to_{\bar\eta_c} r'}{r \to_{\bar\eta} r'} \qquad \frac{r \to_{\bar\eta} r'}{(\lambda xr)\vec{s} \to_{\bar\eta} (\lambda xr')\vec{s}} \qquad \frac{r \to_{\bar\eta} r'}{sr\vec{t} \to_{\bar\eta} sr'\vec{t}}$$

$\bar\eta$ is confluent and strongly normalizing – see [1] for a short proof.

2.6. Typed commutation

We will now establish commutation properties analogous to (2) and (3) for η-expansion and use them to derive strong normalization first of $\bar\eta\theta$ and then of $\beta\bar\eta$.

Proposition 5. $\Delta r \to_{\bar\eta} s \Rightarrow s = \Delta' r'$ &

- *either* $\Delta = \Delta'$ *and* $r \to_{\bar\eta} r'$ (*an inner reduction in case* $\hat\Delta$),
- *or* $\Delta \to_{\bar\eta} \Delta'$ *and* $r = r'$.

Proof. Induction on Δ. *Case* $\{\}$. Trivial. *Case* $\lambda x.\Delta_0(\Delta_1 r \Delta_2 x)$. An η-expansion of this term has to have the form

$$\lambda x.\Delta_0(\Delta_1 r \Delta_1 x) \to_{\bar\eta} \lambda xs \quad \text{with } \Delta_0(\Delta_1 r \Delta_2 x) \to_{\bar\eta} s.$$

By induction hypothesis for Δ_0 the term s has the form $\Delta_0' s'$ and there are two possibilities: either $\Delta_0 \to_\eta \Delta_0'$ and $s' = \Delta_1 r \Delta_2 x$ – in this case we can choose $\Delta' := \lambda x.\Delta_0'(\Delta_1\{\}\Delta_2 x)$ and obtain $\Delta \to_{\bar\eta} \Delta'$ (note that any η-expansion on a variable can also be performed on an application). Or $\Delta_0 = \Delta_0'$ and $\Delta_1 r \Delta_2 x \to_{\bar\eta} s'$.

- If $\Delta_1 r\Delta_2 x \to_{\bar\eta_c} \lambda z.(\Delta_1 r\Delta_2 x)z$ then by induction hypothesis $\Delta_0 = \{\}$, so we can set $\Delta' := \lambda x\lambda z.(\Delta_1\{\}\Delta_2 x)z$ and get $\Delta \to_{\bar\eta} \Delta'$.
- Otherwise the expansion might stem from an inner expansion of $\Delta_1 r$. By induction hypothesis this is either a reduction $\Delta_1 r \to_{\bar\eta} \Delta_1' r$ and we set $\Delta' := \lambda x.\Delta_1'\{\}\Delta_2 x$, or a reduction $\Delta_1 r \to_{\bar\eta} \Delta_1 r'$ (an inner one, if $\Delta_1 \neq \{\ \}$), in which case we are obviously done.
- Finally, the expansion could stem from $\Delta_1 r\Delta_2 x \to_{\bar\eta} \Delta_1 r\Delta_2' x$. Then we set $\Delta' := \lambda x.\Delta_1\{\}\Delta_2' x$. $\square$

Lemma 6.

(4) $\delta \triangleright \bar\eta \subset \bar\eta \triangleright \delta$,

(5) $\beta \cap \eta \triangleright \bar\eta \subset \bar\eta \triangleright \beta \cap \eta$.

Proof. (4) Assume $r \to_\delta s \to_{\bar\eta} t$. We show $\exists s'.r \to_{\bar\eta} s' \to_\delta t$ by induction on r.

Case $(\Delta r)s \to_{\delta_c} (\Delta r)@s \to_{\bar\eta} t$. Note that Δ is necessarily of the form $\lambda x.\Delta_0(\Delta_1\{\}\ \Delta_2 x)$, so $(\Delta r)@s = \Delta_0((\Delta_1 r)\Delta_2 s) \to_{\bar\eta} t$. Using the previous proposition, this reduction stems either from $\lambda x.\Delta_0((\Delta_1 r)(\Delta_2 x)) \to_{\bar\eta} \lambda x.\Delta_0'((\Delta_1' r')(\Delta_2' x))$ or from $s \to_{\bar\eta} s'$. Taking both cases together we can reorder the reductions to

$$(\lambda x.\Delta_0((\Delta_1 r)(\Delta_2 x)))s \to_{\bar\eta} (\lambda x.\Delta_0'((\Delta_1' r')(\Delta_2' x)))s' \to_\delta t.$$

Case $r \to_\delta s \to_{\bar\eta_c} \lambda x.sx$ with an inner reduction $r \to_\delta s$. Since s cannot be an abstraction, neither can r. Therefore we can reduce $r \to_{\bar\eta} \lambda x.rx \to_\delta \lambda x.sx$.

Case both reductions are inner ones. If both reductions happen in the same subterm, we can use the induction hypothesis. Otherwise the reductions are surely independent and can be reordered without difficulties.

(5) Assume $r \to_{\beta\cap\eta} s \to_{\bar\eta} t$. The basic structure of the proof is just as for (4), i.e., we essentially only have to cover the cases where either reduction is a contraction.

Case $\lambda x.(\lambda xr)x \to_{\eta_c} \lambda xr \to_{\bar\eta} \lambda xr' : \lambda x.(\lambda xr)x \to_{\bar\eta} \lambda x.(\lambda xr')x \to_{\eta_c} \lambda xr'$.

Case $(\lambda x.rx)s \to_{\beta\cap\eta} rs \to_{\bar\eta_c} \lambda z.rsz : (\lambda x.rx)s \to_{\bar\eta_c} \lambda z.(\lambda x.rx)sz \to_{\beta\cap\eta} \lambda z.rsz$.

Case $(\lambda x.rx)s \to_{\beta\cap\eta} rs \to_{\bar\eta} r's' : (\lambda x.rx)s \to_{\bar\eta} (\lambda x.r'x)s' \to_{\beta\cap\eta} r's'$.

Case $rs \to_{\beta\cap\eta} r's' \to_{\bar\eta_c} \lambda x.r's'x : rs \to_{\bar\eta_c} \lambda x.rsx \to_{\beta\cap\eta} \lambda x.r's'x$. $\square$

Theorem 7. $\bar\eta\theta \Downarrow$.

Proof. Show $r \Downarrow_{\bar\eta\theta}$ by induction on $r \Downarrow_{\bar\eta}$ and side induction on $r \Downarrow_\beta$. Let $r \to_{\bar\eta\theta} r'$.

Case $r \to_{\bar\eta} r'$: this reduces the expansion height, so $r' \Downarrow_{\bar\eta\theta}$ by the main induction hypothesis.

Case $r \to_\delta r'$: we claim that $\#_{\bar\eta} r' \leq \#_{\bar\eta} r$. So let $r' \to_{\bar\eta} r''$. Then $r \to_{\bar\eta} s \to_\delta r''$ by commutation property (4), so $s \Downarrow_{\bar\eta\theta}$ by the main induction hypothesis. If follows that $r'' \Downarrow_{\bar\eta}$ with $\#_{\bar\eta} r'' < \#_{\bar\eta} r$. Thus we can use the main or side induction hypothesis for r' to conclude $r' \Downarrow_{\bar\eta\theta}$.

Case $r \to_{\beta\cap\eta} r'$: similar, using property (5). $\square$

Corollary 8. $\beta\bar\eta \Downarrow$.

Proof. Show $r \Downarrow_{\beta\bar\eta}$ by induction on $r \Downarrow_\nu$ with side induction on $r \Downarrow_{\bar\eta\theta}$. So let $r \to_{\beta\bar\eta} r'$.

- In case of a strict β-reduction $r \rightarrow_{\beta!} r'$, the result has less v-height and is therefore strongly normalizing by induction hypothesis.
- In case of an $\bar{\eta}$-expansion, we claim that $\#_v r' \leq \#_v r$. So let $r' \rightarrow_v r''$. Then $r_\eta \leftarrow r' \rightarrow_v r''$ and therefore $\#_v r'' < \#_v r$. We can thus apply the main or side induction hypothesis to r'.
- In case of a $\beta \cap \eta$- or δ-reduction we similarly invoke the induction hypotheses. $\square$

3. Raw terms of pure type systems

Since the basic commutation results apply to the untyped λ-calculus, it is conceivable that they adapt to the raw term language of Pure Type Systems. In this section we highlight the difficulties in the adjustment of Sections 2.2–2.4, which mainly stem from reductions in abstracted types. As reduction on raw terms is not confluent in general, we cannot establish all commutation properties that held for the untyped λ-calculus and need to postpone them to the next section.

3.1. Basic definitions

The grammar of *raw terms* is given by

$$r, s, A, B \ ::= \ x \,|\, rs \,|\, \lambda x : A.r \,|\, \Pi x : A.B \,|\, \mathfrak{s}.$$

Here $\mathfrak{s}$ ranges over a given set of sorts S. $\lambda x : A.r$ and $\Pi x : A.r$ bind x in r. We adopt most of the notational conventions discussed in Section 2.1 and abbreviate $A \rightarrow B := \Pi x : A.B$ (x new).

The set $\mathrm{AV}r \subseteq \mathrm{FV}r$ of *abstracted variables* of r is defined recursively by

$$\mathrm{AV}x := \mathrm{AV}\mathfrak{s} := \emptyset, \qquad \mathrm{AV}\lambda x : A.r := \mathrm{FV}A \cup (\mathrm{AV}r \setminus \{x\}),$$
$$\mathrm{AV}(rs) := \mathrm{AV}r \cup \mathrm{AV}s, \quad \mathrm{AV}\Pi x : A.B := \mathrm{FV}A \cup (\mathrm{AV}B \setminus \{x\}).$$

Abstracted variables are free variables that occur in abstracted types. This concept will be needed for the analysis of substitution in the next section.

Reduction. *β- and η-contraction* on raw terms are given by

$$(\lambda x : A.r)s \rightarrow_{\beta_c} r[x := s]. \qquad \lambda x : A.rx \rightarrow_{\eta_c} r \quad \text{if } x \notin \mathrm{FV}r.$$

The relations β and η are built using the standard term closure.

3.2. $\beta \cap \eta$-Reduction

Where we could simply employ the intersection $\beta \cap \eta$ in the case of the λ-calculus, we face a well-known difficulty in general Pure Type Systems – $\beta\eta$ is not confluent on raw terms, as Nederpelt's counter example shows:

$$\lambda x : A.x\,{}_\beta\!\leftarrow \lambda x : A.(\lambda y : B.y)x \rightarrow_\eta \lambda y : B.y.$$

Such divergences can only be joined in a typed setting, using uniqueness of types. In order to treat such β-reductions separately we define *$\hat{\beta}$-contraction* by

$$\lambda x : A.(\lambda x : B.r)x \rightarrow_{\hat{\beta}_c} \lambda x : A.r,$$

and let $\hat{\beta}$ be the full term closure of $\hat{\beta}_c$. Note that $\hat{\beta}$-redexes are also η-redexes.

Remark. $\hat{\beta}$ is the only reduction that we cannot properly commute with $\underline{\eta}$ due to the above counterexample. In well-behaved (e.g., functional) Pure Type Systems, however, we can use uniqueness of types and the Church–Rosser property of $\beta\eta$ to get commutation, as we will see below.

3.3. Extended $\eta \cap \beta_c$-contraction

As in Section 2.2 we define Δ by

$$\delta \ni \Delta ::= \lambda x : A.\Delta_0\{\Delta_1\{\}\Delta_2\{x\}\} \mid \{\}.$$

Again, $\hat{\Delta}$ denotes a $\Delta \neq \{\}$ and $(\Delta r)@s$ is defined only for $\hat{\Delta}$:

$$(\lambda x : A.\Delta_0(\Delta_1 r \Delta_2 x))@s := \Delta_0(\Delta_1 r \Delta_2 s).$$

Extended contraction is given by $(\Delta r)s \to_{\Delta_c} (\Delta r)@s$. δ_c is the union of all Δ_c. δ denotes the term closure of δ_c. As in Section 2.2 we have $\delta \subset \beta$ and $\hat{\Delta} r \to_{\eta}^{+} r$.

Proposition 9. *$\Delta r \to_{\eta\underline{\eta}} s \Rightarrow$*

- *$s = \Delta r'$ with $r \to_{\eta\underline{\eta}} r'$ or*
- *$s = \Delta' r$ with $\Delta \to_{\eta\underline{\eta}} \Delta'$.*

3.4. Restricted term closure

One new aspect of raw terms as compared to normal λ-terms is the domain A of λ-abstractions $\lambda x : A.r$. Just as δ and $\hat{\beta}$, reductions in such abstracted types is non-computational. In order to isolate such reductions, we write ϕ' for the restricted term closure of ϕ_c, obtained by leaving out the rule

$$\frac{A \to_{\phi} A'}{\lambda x : A.r \to_{\phi} \lambda x : A'.r}$$

and set $\tilde{\phi} := \phi \setminus \phi'$ (“reduction in abstracted types”).

3.5. θ-Reduction

Just as in Section 2, θ shall comprise all non-computational β-reductions, so

$$\theta := \hat{\beta}'\delta'\tilde{\beta}, \quad \beta! := \beta \setminus \theta.$$

Since $\hat{\beta}$ and δ are subsets of β we get $\beta = \beta!\theta$.

Remark. Note that we used the restricted term closure $\hat{\beta}'$ of $\hat{\beta}$, because $\hat{\beta} \setminus \hat{\beta}' \subset \tilde{\beta}$ (and similarly for δ).

Proposition 10.

(1) $\underline{\eta} \triangleright \beta! \subset \beta! \triangleright \underline{\eta}^{*}$,

(2′) $\underline{\eta}' \triangleright \tilde{\beta} \subset (\varepsilon\tilde{\beta}) \triangleright \underline{\eta}'$,

(3) $\underline{\eta} \rhd \eta \subset (\eta \rhd \underline{\eta})\varepsilon$,

(4) $\underline{\tilde{\beta}} \rhd \beta! \subset \beta! \rhd \underline{\tilde{\beta}}^*$,

(5′) $\underline{\tilde{\beta}} \rhd \delta' \subset \delta' \rhd \underline{\tilde{\beta}}^*$.

Proof. Given a divergence of the respective form which is starting from r, we show the claims independently by induction on r.

(1) The proof is analogous to that of Lemma 2 in Section 2.3(1), with all new cases being trivial.

(2′) Note that we only exchange η'-reductions which do not take place in abstracted types with β-reductions in abstracted types. The case $r_{\eta'} \leftarrow \lambda x : A.rx \rightarrow_{\tilde{\beta}} \lambda x : A'.rx$ requires the possibility that the left joining reduction sequence is empty.

(3),(4),(5′) Simple, because all dangerous cases are excluded. □

Remark. We cannot yet provide a reasonable commutation property between $\underline{\eta}$ and $\hat{\beta}$ or $\underline{\tilde{\eta}}$ and $\tilde{\beta}$ for the case of raw terms. In the next section we will use uniqueness of types to derive such properties and also stronger variants of (2′) and (5′).

4. η-Expansion in Pure Type Systems

We follow [3] in the presentation of η-expansion for Pure Type Systems, adopting its restriction of $\bar{\eta}_c$ to Π-types in β-normal form. Then we use confluence for $\beta\eta$ on typed terms to derive the lacking commutation properties. Finally, we provide typing derivations for δ-reduction from which strong normalization follows.

4.1. Pure Type Systems

A *functional Pure Type System* is determined by three data: A set S of *sorts*, a partial function $\mathcal{A} \subseteq S \times S$ *(axioms)* and a partial function $\mathcal{R} \subset S^3$ *(rules)*.[7]

A *context* Γ is an expression of the form $x_1 : A_1, \ldots, x_n : A_n$ with disjoint $\vec{x}$ (the domain of the context, denoted by domΓ). The empty context will be omitted, if reasonable. The notation $\Gamma, x : A$ is used iff $x \notin \mathrm{dom}\Gamma$.

The rules of Pure Type Systems with $\beta\eta$-equality (see Fig. 1) derive judgments of the form $\Gamma \vdash r : A$ with r a raw term.

A *term* is a raw term typable in this system. From now on r, s, t, A, B, C will range over terms and Γ over *legal contexts*, i.e., over $\{\Gamma \mid \exists r, A.\Gamma \vdash r : A\}$.

We will presuppose most of the standard properties of functional Pure Type Systems (such as subject reduction, generation, uniqueness of types, strengthening, etc.) and refer to [4,7] for details.

Furthermore *we assume strong normalization of* β, in order to be able to use results of [3], such as the Church–Rosser-property for $\beta\eta$ and strong normalization of $\beta\eta$ [7].

[7] We restrict to *functional* Pure Type Systems in order to guarantee uniqueness of types.

$$\frac{(C, s) \in \mathcal{A}}{\vdash C : s} \qquad \frac{\Gamma \vdash r : A \qquad \Gamma \vdash A' : s \qquad A =_{\beta\eta} A'}{\Gamma \vdash r : A'}$$

$$\frac{\Gamma \vdash r : A \qquad \Gamma \vdash B : s}{\Gamma, x : B \vdash r : A} \qquad \frac{\Gamma, x : A \vdash r : B \qquad \Gamma \vdash \Pi x : A.B : s}{\Gamma \vdash \lambda x : A.r : \Pi x : A.B}$$

$$\frac{\Gamma \vdash A : s}{\Gamma, x : A \vdash x : A} \qquad \frac{\Gamma \vdash r : \Pi x : A.B \qquad \Gamma \vdash s : A}{\Gamma \vdash rs : B[x := s]}$$

$$\frac{\Gamma \vdash A : s_1 \qquad \Gamma, x : A \vdash B : s_2}{\Gamma \vdash \Pi x : A.B : s_3} (s_1, s_2, s_3) \in \mathcal{R}$$

Fig. 1. Rules of pure type systems with $\beta\eta$-equality.

4.2. *Commutation from confluence of $\beta\eta$*

Using the Church–Rosser-property for $\gamma := \beta\eta$, we can show the promised confluence property for $\hat{\beta}$.

Proposition 11. (6′) $\underline{\eta'} \triangleright \hat{\beta} \ \subset \ \hat{\beta}\tilde{\gamma}^* \triangleright \underline{\tilde{\gamma}}^* \underline{\eta}$.

Proof. The case where we invoke confluence of γ is the only one worth mentioning: so assume

$$\lambda x : B.r_{\eta_c} \leftarrow \lambda x : A.(\lambda x : B.r)x \rightarrow_{\hat{\beta}_c} \lambda x : A.r.$$

The Church–Rosser property for γ yields a C with

$$B \rightarrow_{\gamma^*} C_{\tilde{\gamma}^*} \leftarrow A,$$

so we get $\lambda x : B.r \rightarrow_{\tilde{\gamma}^*} \lambda x : C.r_{\tilde{\gamma}^*} \leftarrow \lambda x : A.r.$ $\Box$

Using the same technique we can now eliminate the restrictions on reductions in (2′), (5′), (6′) and derive one further commutation property needed in the proof of theorem 13 in Section 4.3.

Lemma 12.

(2) $\underline{\eta} \triangleright \tilde{\beta} \subset \tilde{\beta}\tilde{\gamma}^* \triangleright \tilde{\gamma}^* \underline{\tilde{\beta}}^*$,

(5) $\underline{\tilde{\beta}} \triangleright \delta \subset \delta'\tilde{\gamma}^* \triangleright \underline{\tilde{\gamma}}^*$,

(6) $\underline{\eta} \triangleright \hat{\beta} \subset \hat{\beta}\tilde{\gamma}^* \triangleright \underline{\tilde{\gamma}}^* \underline{\eta}$,

(7) $\underline{\eta} \triangleright \delta \subset (\delta\tilde{\gamma}^* \triangleright \underline{\tilde{\gamma}}^* \underline{\eta})\varepsilon$.

4.3. ν-Reduction

We continue as in Section 2.4, defining

$$\nu := (\underline{\eta\tilde{\beta}}\theta\tilde{\eta})^* \triangleright \beta!.$$

A ν-step consists in one strict β-reduction, preceded by a few backward $\eta\tilde{\beta}$ or forward $\theta\tilde{\eta}$-steps.

Theorem 13. $\nu \subset (\beta\eta)^+ \triangleright \underline{\eta}^*$.

Proof. Using the commutation properties (1)–(7), similar to the proof of theorem 3 in Section 2.4. □

Corollary 14. $r \Downarrow_\nu$.

Proof. Like the proof of Corollary 4 in Section 2.4, using the theorem and strong normalization of $\beta\eta$. □

4.4. η-Expansion

As in the simply typed case the definition of η-expansion is only reasonable in a typed context. Therefore $\Gamma \vdash r \mapsto_{\bar{\eta}} r'$ will be defined under the assumption that r is typable in Γ (see below).

Counter examples (first exhibited in [9]) show that it is not appropriate to expand a non-abstraction whenever it gets a Π-type: consider $A := (\lambda z : y \to y.y)x =_\beta y$. In the context $\Gamma := y : *, x : y \to y$ we can type $x : \Pi z : A.y$; a putative expansion would lead to the infinite reduction sequence

$$\Gamma \vdash x \to_{\bar{\eta}_c} \lambda z : A.xz \to_{\bar{\eta}} \lambda z : A[x := \lambda z : A.xz].xz \to_{\bar{\eta}} \ldots$$

To overcome this source of divergence we will follow Barthe [3], who restricted expansions $r \to_{\bar{\eta}_c} \lambda x : A.rx$ to those cases where A is β-normal and r is neutral. *So $\bar{\eta}$-expansion* is defined by

$$\frac{\Gamma \vdash r : \Pi x : A.B \text{ neutral} \qquad \Gamma \vdash \Pi x : A.B \in \mathrm{NF}_\beta}{\Gamma \vdash r \to_{\bar{\eta}_c} \lambda x : A.rx}.$$

$$\frac{\Gamma \vdash r \to_{\bar{\eta}_c} r'}{\Gamma \vdash r \to_{\bar{\eta}} r'} \qquad \frac{\Gamma, x : A \vdash B \to_{\bar{\eta}} B' \qquad \Gamma \vdash \Pi x : A.B : C}{\Gamma \vdash \Pi x : A.B \to_{\bar{\eta}} \Pi x : A.B'}$$

$$\frac{\Gamma, x : A \vdash r \to_{\bar{\eta}} r' \qquad \Gamma \vdash (\lambda x : A.r)\vec{s} : B}{\Gamma \vdash (\lambda x : A.r)\vec{s} \to_{\bar{\eta}} (\lambda x : A.r')\vec{s}}$$

$$\frac{\Gamma \vdash s \to_{\bar{\eta}} s' \qquad \Gamma \vdash rs\vec{s} : A}{\Gamma \vdash rs\vec{t} \to_{\bar{\eta}} rs'\vec{t}} \qquad \frac{\Gamma \vdash A \to_{\bar{\eta}} A' \qquad \Gamma \vdash \Pi x : A.B : C}{\Gamma \vdash \Pi x : A.B \to_{\bar{\eta}} \Pi x : A'.B}$$

Fig. 2. Rules of type-restricted term closure.

The reduction rules for η-expansion are obtained by the rules of Fig. 2. Note that $\Gamma \vdash r \to_{\bar{\eta}} r'$ implies $\Gamma \vdash r : A$ for a certain A. We use $\bar{\eta}^{\Gamma}$ for the set of pairs (r, r') with $\Gamma \vdash r \to_{\bar{\eta}} r'$ and let $\bar{\eta}$ be given by the union of all $\bar{\eta}^{\Gamma}$ with Γ legal. *Inner expansion* $\bar{\eta}_i$ is defined in the obvious way.

Remarks.

1. Ghani's definition [9] of $\bar{\eta}$ required $\beta\bar{\eta}$-normality of the type in the expansion rule. This had several drawbacks, the most important one being that the set of $\bar{\eta}$-reducts was not decidable.
2. If $\Gamma \vdash r \to_{\bar{\eta}} r'$ then $r' \to_{\eta} r$. We note without proof (referring the interested reader to [3]) that $\to_{\bar{\eta}}$ enjoys subject reduction, i.e.,

$$\Gamma \vdash r : A \;\&\; \Gamma \vdash r \to_{\bar{\eta}} r' \;\Rightarrow\; \Gamma \vdash r' : A.$$

   Thanks to subject reduction, typing and reduction judgments can be merged for the sake of shorter notation: e.g., $\Gamma \vdash r \to_{\beta} r' \to_{\bar{\eta}} r'' : A$ stands for the three separate statements $\Gamma \vdash r : A \;\&\; r \to_{\beta} r' \;\&\; \Gamma \vdash r \to_{\bar{\eta}} r'$.
3. Barthe [3] has proved that $\bar{\eta}^{\Gamma}$ is strongly normalizing in β-normalizing Pure Type Systems, i.e., $\Gamma \vdash r : A \;\Rightarrow\; r \Downarrow_{\bar{\eta}^{\Gamma}}$.
4. In the sequel publication [4], Barthe conjectures that $\beta\bar{\eta}$, where $\bar{\eta}$ uses the full term closure, is strongly normalizing in the Calculus of Constructions. The verification of essential properties of his model construction has some gaps which seem difficult to fill, although the claim itself is reasonable. In the present work, we concentrate on the restricted version of reduction, for which a purely syntactic proof can be given.
5. The following property of η-expansion will be required later and is shown by a simple induction.

$$x \notin \mathrm{AV}r \;\&\; \Gamma \vdash r \to_{\bar{\eta}} r' \;\Rightarrow\; \Gamma \vdash r[x := s] \to_{\bar{\eta}} r'[x := s] \quad \text{if } s \text{ is neutral.}$$

4.5. Types for Δ

Following the structure of Section 2 we now set out the exchange of $\to_{\theta}$ with a following η-expansion. We first derive types for Δ. To this end we define $\Gamma \vdash \Delta : A$ inductively by

$$\frac{\Gamma \vdash r : A}{\Gamma \vdash \{\} : A} \qquad \frac{\Gamma \vdash \Delta_0 : B \quad \Gamma \vdash \Delta_1 : \Pi x : A.B \quad \Gamma \vdash \Delta_2 : A}{\Gamma \vdash \lambda x : A.\Delta_0(\Delta_1\{\}\Delta_2 x) : \Pi x : A.B}$$

Proposition 15. $\Gamma \vdash \Delta : A \;\&\; \Gamma \vdash r : A \;\Rightarrow\; \Gamma \vdash \Delta r : A$.

Proof. Construct the derivation recursively along that of $\Gamma \vdash \Delta : A$. $\square$

Corollary 16.

(i) $\Gamma \vdash r \to_{\bar{\eta}_i} r' : A \;\&\; \Gamma \vdash \Delta : A \;\Rightarrow\; \Gamma \vdash \Delta r \to_{\bar{\eta}_i} \Delta r'$,
(ii) $\Gamma, x : A \vdash \Delta x \to_{\bar{\eta}} \Delta' x \;\&\; \Gamma \vdash r : A \;\Rightarrow\; \Gamma \vdash \Delta r \to_{\bar{\eta}} \Delta' r$, *provided* $x \notin FV\Delta$ *and* r *neutral.*

Proof. T he first claim follows easily from type correctness of Δ. The second is an instance of Remark 5 above, using $x \notin \mathrm{AV}\Delta x$. $\square$

4.6. Typed commutation properties

The rest of the strong normalization proof proceeds completely parallel to that of Section 2.6.

Proposition 17. $\Gamma \vdash \Delta r \to_{\bar\eta} s : A \Rightarrow s = \Delta' r'$ &

- *either* $\Delta = \Delta'$ *and* $\Gamma \vdash r \to_{\bar\eta} r'$ ($\to_{\bar\eta_i}$ *in case* $\hat\Delta$),
- *or* $\Gamma, x : A \vdash \Delta x \to_{\bar\eta} \Delta' x$ *and* $r = r'$.

Proof. Induction on Δ. *Case* $\{\}$. Trivial.

Case $\Delta = \lambda x : A.t := \lambda x : A.\Delta_0(\Delta_1\{\ \}\Delta_2 x)$. There are two possibilities for $\Delta r \to_{\bar\eta} s$: either it has been concluded from $A \to_{\bar\eta} A'$ and the claim is obvious, or it stems from $\Gamma, x : A \vdash t \to_{\bar\eta} s : B$, i.e., from $\Gamma, x : A \vdash \Delta_0(\Delta_1 r \Delta_2 x) \to_{\bar\eta} s : B$. By induction hypothesis for Δ_0 the term s has either of the following forms:

- $\Delta_0'(\Delta_1 r \Delta_2 x)$ with $\Gamma, x : A, y : B \vdash \Delta_0 y \to_{\bar\eta} \Delta_0' y$: $\Gamma, x : A \vdash \Delta_1 r \Delta_2 x : B$, so the corollary yields $\Gamma, x : A \vdash \Delta_0(\Delta_1 r \Delta_2 x) \to_{\bar\eta} \Delta_0'(\Delta_1 r \Delta_2 x)$.
- $\Delta_0 t'$ with $\Gamma, x : A \vdash \Delta_1 r \Delta_2 x \to_{\bar\eta} t'$. Using the induction hypothesis there are three possibilities for such a reduction:
  - $\Gamma, x : A \vdash \Delta_1 r \Delta_2 x \to_{\bar\eta_c} \lambda y : C.(\Delta_1 r \Delta_2 x) y$. In this case we set $\Delta' := \lambda x : A.\Delta_0 \lambda y : C.(\Delta_1\{\}\Delta_2 x) y$ and use corollary (ii).
  - $\Gamma, x : A \vdash \Delta_1 r \Delta_2 x \to_{\bar\eta} \Delta_1' r \Delta_2 x$. The claim follows easily from the induction hypothesis for Δ_1 and corollary (i).
  - $\Gamma, x : A \vdash \Delta_1 r \Delta_2 x \to_{\bar\eta} \Delta_1 r \Delta_2' x$. We set $\Delta' := \lambda x : A.\Delta_0(\Delta_1\{\}\Delta_2' x)$. $\square$

Lemma 18.

(a) $\delta' \rhd \bar\eta \subset \bar\eta \rhd \delta'$,

(b) $\hat\beta' \rhd \bar\eta \subset \bar\eta \rhd \hat\beta'$,

(c) $\tilde\beta \rhd \bar\eta \subset \bar\eta \rhd \tilde\beta$.

Proof. Assume $\Gamma \vdash r \to_\phi s \to_{\bar\eta} t$ with $\phi \in \{\delta', \hat\beta', \tilde\beta\}$. We show the respective claims independently by induction on r. The case that the reductions occur in different subterms allows a simple re-ordering. In case the reductions take place in the same subterm, we use the induction hypothesis. In the remaining cases at least one of the reductions is an outer reduction.

(a) *Case* $\Gamma \vdash (\Delta r)s \to_{\delta_c} (\Delta r)@s \to_{\bar\eta} t$. Δ has the form $\lambda x : A.\Delta_0(\Delta_1\{\}\Delta_2 x)$, so the reduction reads

$$\Gamma \vdash (\lambda x : A.\Delta_0(\Delta_1\{\}\Delta_2 x))s \to_{\delta_c} \Delta_0(\Delta_1 r \Delta_2 s) \to_{\bar\eta} t.$$

We use the previous proposition and the definition of $\to_{\bar\eta}$ to obtain one of the following subcases.

Subcase $\Gamma, x : A, y : B \vdash \Delta_0 y \to_{\bar\eta} \Delta'_0 y$:

$$\dfrac{\dfrac{\Gamma, x : A, y : B \vdash \Delta_0 y \to_{\bar\eta} \Delta'_0 y}{\Gamma, x : A \vdash \Delta_0(\Delta_1 r \Delta_2 x) \to_{\bar\eta} \Delta'_0(\Delta_1 r \Delta_2 x)}\,(ii)}{\Gamma \vdash \lambda x : A.b\Delta_0(\Delta_1 r \Delta_2 x)s \to_{\bar\eta_i} \lambda x : A.\Delta'_0(\Delta_1 r \Delta_2 x)s}$$

Subcase $\Gamma \vdash \Delta_1 r \to_{\bar\eta_i} \Delta'_1 r'$.

$$\dfrac{\dfrac{\Gamma \vdash \Delta_1 r \to_{\bar\eta_i} \Delta'_1 r'}{\Gamma, x : A \vdash (\Delta_1 r)\Delta_2 x \to_{\bar\eta_i} (\Delta'_1 r')\Delta_2 x}}{\Gamma \vdash \lambda x : A.\Delta_0(\Delta_1 r \Delta_2 x)s \to_{\bar\eta_i} \lambda x : A.\Delta_0(\Delta'_1 r' \Delta_2 x)s}$$

Subcase $\Gamma \vdash \Delta_2 s \to_{\bar\eta} \Delta'_2 s$ with $\Gamma, x : A \vdash \Delta_2 x \to_{\bar\eta} \Delta_2 x$.

$$\dfrac{\dfrac{\dfrac{\Gamma, x : A \vdash \Delta_2 x \to_{\bar\eta} \Delta_2 x}{\Gamma, x : A \vdash \Delta_1 r \Delta_2 x \to_{\bar\eta_i} \Delta_1 r \Delta'_2 x}}{\Gamma, x : A \vdash \Delta_0(\Delta_1 r \Delta_2 x) \to_{\bar\eta_i} \Delta_0(\Delta_1 r \Delta'_2 x)}\,(i)}{\Gamma \vdash (\lambda x : A\Delta_0(\Delta_1 r \Delta_2 x))s \to_{\bar\eta_i} (\lambda x : A.\Delta_0(\Delta_1 r \Delta'_2 x))s}$$

Subcase $\Gamma \vdash s \to_{\bar\eta} s'$.

$\Gamma \vdash (\lambda x : A\Delta_0(\Delta_1 r \Delta_2 x))s \to_{\bar\eta_i} (\lambda x : A.\Delta_0(\Delta_1 r \Delta'_2 x))s.$

In all subcases we can append the δ-reduction.

(b) The proof of Lemma 6 in Section 2.6 can be reused without essential changes.

(c) This is very simple: as already mentioned, we have to deal with the cases of an outer reduction on either side, and both possibilities lead to trivial exchanges. □

Theorem 19. $\Gamma \vdash r : A \Rightarrow r \Downarrow_{\bar\eta^\Gamma \theta}$.

Proof. Induction on $r \Downarrow_{\bar\eta^\Gamma}$, side induction on $r \Downarrow_\beta$. The proof is similar to that of Theorem 7 in Section 2.6. □

Corollary 20. $\Gamma \vdash r : A \Rightarrow r \Downarrow_{\beta\bar\eta^\Gamma}$.

Proof. Induction on $r \Downarrow_v$ and side induction on $r \Downarrow_{\bar\eta^\Gamma \theta}$, as in Section 2.6. □

More verbosely, the statement of the corollary is
In every functional Pure Type System with $\beta\eta$-equality that enjoys strong normalization for $\beta\eta$-reduction, the combination of β-reduction and η-expansion (restricted to not occur in abstracted types) is strongly normalizing.

4.7. Simulation

In the final step we use a simulation method to overcome the remaining restriction of the last corollary, namely that reduction shall not occur in abstracted types.

To this end we translate typed terms in a given Pure Type System by expanding every abstraction as follows:[8]

$$[[\lambda x : A.r]] := (\lambda y : \mathfrak{s}.\lambda x : y.[[r]])[[A]], \quad y \text{ new},$$

where $\mathfrak{s}$ is the sort of A (this is uniquely determined by the type of $\lambda x : A.r$). All other term constructors are translated homomorphically.

Note that we need to be able to type the abstraction $\lambda y : \mathfrak{s}.(\ldots)$. This is always possible if the ruleset of the Pure Type System in regard has the following closure property

$$\frac{(\mathfrak{s}_1, \mathfrak{s}_2, \mathfrak{s}_3) \in \mathcal{R}}{(\mathfrak{s}_1, \mathfrak{s}_3, \mathfrak{s}_3) \in \mathcal{R}}. \tag{‡}$$

To see this let assume a typing assignment to $\lambda x : A.r$:

$$\frac{\Gamma, x : A \vdash r : B \quad \Gamma \vdash \Pi x : A.B : \mathfrak{s}_3}{\Gamma \vdash \lambda x : A.r : \Pi x : A.B}$$

By the generation lemma there exist $\mathfrak{s}_1, \mathfrak{s}_2$ such that

$$\Gamma \vdash A : \mathfrak{s}_1, \qquad \Gamma \vdash B : \mathfrak{s}_2, \qquad \Gamma \vdash \Pi x : A.B : \mathfrak{s}_3$$

with $(\mathfrak{s}_1, \mathfrak{s}_2, \mathfrak{s}_3) \in \mathcal{R}$. By (‡) we obtain $\Gamma \vdash \Pi y : \mathfrak{s}_1.\Pi x : A.B : \mathfrak{s}_3$ and can thus type the abstraction $\lambda y : \mathfrak{s}_1.\lambda x : A.r$.

The property (‡) holds for many interesting example systems and in particular for the Calculus of Constructions.

Proposition 21. $\Gamma \vdash r \rightarrow_{\beta\bar{\eta}} r' \Rightarrow \Gamma \vdash [[r]] \rightarrow_{(\beta\bar{\eta})^+} [[r']]$

Proof. Simple induction along the definition of $\rightarrow_\beta$ and $\Gamma \vdash r \rightarrow_{\bar{\eta}} r' : A$. $\square$

Corollary 22. *In every functional Pure Type System with $\beta\eta$-equality that enjoys strong normalization for $\beta\eta$-reduction and fulfills (‡), the combination of β-reduction and η-expansion is strongly normalizing.*

5. Conclusions and further work

We have presented a simple characterization of the particular forms of β-reductions which are provoked by η-expansion $\bar{\eta}$. This has served to derive a normalization theorem for combined $\beta\bar{\eta}$-reduction relative to strong normalization for β and $\bar{\eta}$ alone. The underlying analysis of commutation properties is original and yields new insight into the computational behavior of η-expansion.

However, strong normalization has only been proved for a reduction relation that forgoes expansions in abstracted types and this restriction can only be overcome by reduction simulation that requires the closure property (‡). Although this suffices for the Calculus of Constructions and

[8] Thanks to Gilles Barthe for pointing this possibility out to me.

thus extends results of Ghani in [9], the proof seems unnecessarily complicated and technically demanding. It would be very interesting to find more direct solutions of the strong normalization problem that eliminate the need for (‡) and provide more intuitive insight into the behaviour of η-expansion in Pure Type Systems.

Obviously, the framework of Pure Type Systems is very natural to approach η-expansion in dependently typed systems, reducing the technical complexity of strong normalization arguments by their syntactic simplicity. A natural direction for future research is to explore how the results of this article extend to Σ-types, W-types and universes, as used in Martin–Löf's Type Theory and other dependently typed systems. This also raises the question of how the more syntactically oriented methods of rewriting used in this article compare to the semantical approaches for deciding extensional equality in the respective systems. It should be mentioned, however, that strong normalization of $\beta\bar{\eta}$ is not addressed by such ventures, while it is still necessary in many applications (such as theorem provers and unification) to allow arbitrary (in particular not completely normalizing) rewriting.

Acknowledgments

Gilles Barthe provided valuable discussions on the properties of domain-free Pure Type Systems. I also would like to thank the anonymous referees for their helpful remarks.

References

[1] Klaus Aehlig, Felix Joachimski, Operational aspects of normalization by evaluation, MSCS (2001), submitted.
[2] Yohji Akama, On Mints' reductions for ccc-calculus, in: Proc. 1st TLCA, 1993 (Utrecht), Lecture Notes in Computer Science, vol. 664, Springer, Berlin, 1999, pp. 1–12.
[3] G. Barthe, Existence and uniqueness of normal forms in pure type systems with $\beta\eta$-conversion, in: G. Gottlob, E. Grandjean, K. Seyr (Eds.), Proc. 12th CSL 1998 (Brno), Lecture Notes in Computer Science, vol. 1584, Springer, Berlin, 1999, pp. 241–259.
[4] G. Barthe, Expanding the cube, in: W. Thomas (Ed.), Proc. 2nd FOSSACS 1999 (Amsterdam), Lecture Notes in Computer Science, vol. 1578, Springer, Berlin, 1999, pp. 90–103.
[5] Roberto di Cosmo, A brief history of rewriting with extensionality, Presented at the International Summer School on Type Theory and Term Rewriting (Glasgow), 1996.
[6] G. Dowek, G. Huet, B. Werner, On the definition of the η-long normal form in type systems of the cube, in: H. Geuvers (Ed.), Informal Proceedings of the Workshop on Types for Proofs and Programs, Nijmegen, The Netherlands, 1993.
[7] Herman Geuvers, Logics and Type systems, Ph.D. Thesis, University of Nijmengen, September 1993.
[8] N. Ghani, Adjoint Rewriting, Ph.D. Thesis, University of Edinburgh, 1995.
[9] N. Ghani, η-Expansions in dependent type theory – the calculus of constructions, in: P. de Groote, J.R. Hindley (Eds.), Proc. 3rd TLCA 1997 (Nancy), Lecture Notes in Computer Science, vol. 1210, Springer, Berlin, 1999, pp. 164–180.
[10] Neil Ghani, η-Expansions in: F^{ω}, in: Dirk van Dalen, Marc Bezem (Eds.), Proc. 10th CSL 1996 (Utrecht), Lecture Notes in Computer Science, vol. 1258, Springer, Berlin, 1999, pp. 182–197.
[11] Felix Joachimski, Ralph Matthes, Short proofs of normalization for the simply-typed λ-calculus, permutative conversions and Gödel's T. Archive of Mathematical Logic 42 (2003) 59–87.
[12] Dag Prawitz, Ideas and results in proof theory, in: J.E. Fenstad (Ed.), Proc. 2nd Scandianvian Logic Symposium, North-Holland, Amsterdam, 1971, pp. 235–307.